An Improved Asymmetric Searchable Encryption Scheme

Qi Wu(✉)

School of Information Technology,
Jiangxi University of Finance and Economics, Nanchang 330032, China
wuqiocjzd@l26.com

Abstract. Data to be uploaded on the cloud storage system will often be encrypted for security reasons. No classic encryption & decryption algorithms however provide any search function. To increase user efficiency, an asymmetric searchable encryption scheme is designed in this paper. This improved scheme aims at fixing such flaws as "trapdoors that can be generated by anyone", "ciphertexts that are tampered with at discretion", "key pairs generated by users", "identities encrypted" and "a useless component". Findings suggest that the proposed scheme, while maintaining the established framework, perfectly resolves all the aforementioned flaws in the previous scheme.

Keywords: Cloud server · Searchable encryption · Asymmetric searchable encryption · Key pair · Trapdoor

1 Introduction

With the data volume growing fast, an increasing number of users now upload and store their data on the cloud servers to reduce local storage pressure. Different from previous C/S(Client/Server) mode, cloud servers are usually deemed "semi-trusted" because there exists a fear that the users' data may be hacked. Therefore, encryption is often used to protect the users' data privacy. Traditional encryption & decryption algorithms, such as DES [1], AES [2], and RSA [3], do not support search over encrypted data. When the data volume is huge, storage or bandwidth will not allow cloud users to retrieve all their stored data and then decrypt them to extract the required parts. It is then necessary to design schemes that support search over ciphertexts. Song et al. [4] initially discussed security properties such as controlled searching, provable secrecy, query isolation, and hidden queries, and then constructed the corresponding searchable encryption schemes. Searchable encryption has ever since attracted wide academic interests. In the light of whether index generation and queries use the same key or not, searchable encryption could be categorized into symmetric searchable encryption and asymmetric searchable encryption [5]. Here, only the latter is concerned.

Taking email systems as an example, Boneh et al. [6] constructed PEKS (Public Key Encryption with Keyword Search) model, designed a scheme based on BDH (Bilinear Diffie-Hellman) assumption and trapdoor permutations respectively, and also proposed a construction using Jacobi symbols. Abdalla et al. [7] analyzed the consistency of

G. Wang et al. (Eds.): SpaCCS 2016, LNCS 10066, pp. 39–44, 2016.
DOI: 10.1007/978-3-319-49148-6_4

PEKS in detail, and proved that schemes proposed in [6] are computationally consistent. They successfully built a different statistically-consistent scheme, providing an approach to converting anonymous IBE (Identity Based Encryption) schemes to secure PEKS ones. Khader [8] constructed a PEKS scheme based on K-resilient IBE, which proved IND-CKA (Semantic Security against Adaptive Chosen Keyword Attack) secure under the standard model. Crescenzo et al. [9] proposed a PEKS scheme based on QRP (Quadratic Residuosity Problem). Hwang et al. [10] constructed the PECK (Public key Encryption with Conjunctive Keyword search) model, and then designed a scheme based on DLDH (Decisional Linear Diffie-Hellman) assumption. Compared with previous schemes, the ciphertext and private key of the new scheme proved to be the shortest. Then the model and scheme are extended to the multi-user circumstances. Baek et al. [11] proposed an improved scheme aiming at three issues such as "refreshing keywords", "removing secure channel", and "processing multiple keywords", which are left unconcerned by schemes proposed in [6]. Rhee et al. [12] pointed out that the capability of adversary in the scheme proposed in [11] is too limited, and instead constructed a reinforced secure model and its corresponding improved scheme. Zhao et al. [13] proposed trapdoor-indistinguishable PEKS. Yang et al. [14] proposed a variant aiming at making up missing computational consistency in the scheme proposed in [8], and promoted the efficiency dramatically. Luo et al. [15] proposed a PEKS scheme to tackle the IF (Integer Factorization) problem. Since users vary frequently under mobile cloud storage, Xia et al. [16] designed a PEKS scheme capable of data sharing and ciphertext modification. Shao et al. [17] designed a PEKS scheme in light of the "uni-sender multi-receiver" circumstance in the medical care area, resolving the problem that ciphertext is too long in previous schemes.

Using Elgamal algorithm, Liu et al. [18] fulfilled a PEKS scheme that provided the function of verifying retrieved data in asymmetric searchable encryption. Many flaws however are found in this scheme through our analyses. This paper thus attempts to fixe these flaws one by one as to obtain an improved scheme.

2 Verifiable Public Key Searchable Encryption Scheme

There are 3 principals in the scheme proposed in [18], namely, data uploader Alice, data retriever Bob and the cloud server. There are 5 algorithms in the scheme proposed in [18], namely, parameter generation, generation of the keyword and encrypted files, trapdoor generation, check, and verification, shown as follows.

(1) Parameter generation

Alice selects a big prime p_1 and a generator g_1 in $Z_{p_1}^*$. Alice selects a random integer x_1 $(0 \leq x_1 \leq p-2)$, calculates $y_1 = g_1^{x_1}$, sets her public key to (p_1, g_1, y_1) and private key to x_1. Similarly, Alice selects a generator g_2 in $Z_{p_2}^*$ and a generator g in Z_p^*, then sets the public key of Bob to (p_2, g_2, y_2) and his private key to x_2, sets the public key of the server to (p, g, y) and its private key to x. Alice selects a hash function $H : \{0,1\}^* \rightarrow Z_p^*$ and sets the encryption algorithm and signature algorithm to Elgamal.

(2) Generation of encrypted files

Suppose Alice wants to send a file M including keyword W to Bob. Alice calculates $H(W)$, selects $r_1, r_2 \in_R Z_p^*$, computes $S_1 = r_1 r_2^{H(W)} \bmod p$, $S_2 = r_1^{H(W)^{-1}} r_2 \bmod p$, sets $S = <S_1, S_2>$.

Alice encrypts M with Bob's public key to obtain ciphertext $C_1 = <y_{11}, y_{12}>$.

Alice signs $H(W)$ with x_1 to obtain $sig = <H(W), r, s>$.

Alice encrypts its identity ID_A with Bob's public key to acquire ciphertext $C_2 = <y_{21}, y_{22}>$.

Alice uploads $S = <S_1, S_2>$ and $D = <C_1, sig, C_2>$ to the server.

(3) Trapdoor generation

When Bob wants to retrieve files including keyword W_1, he computes $H(W_1)$, which is encrypted with the server's public key to acquire $T_w = <y_{31}, y_{32}>$. Then, Bob sends T_w to the server.

(4) Check

After receiving T_w, the server decrypts it to obtain $H(W_1)$. Afterwards, the server checks each $S = <S_1, S_2>$ one by one. Once it satisfies that $S_1 = S_2^{H(W_1)}$, the server sends the corresponding $D = <C_1, sig, C_2>$ to Bob.

(5) Verification

After receiving D, first, Bob decrypts C_2 to acquire the identity of the sender. Then, Bob verifies *sig* with the public key of the sender. If the verification succeeds, Bob decrypts C_1. Otherwise, Bob discards.

Through our analyses, there are several flaws in the scheme, shown as follows.

(1) Trapdoors could be generated by anybody. From the description above, we could see, the trapdoor generation algorithm only uses $H(.)$ and the public key of the server. These two are known by every principal, which implies that anybody could generate the trapdoor. Usually, whether in symmetric searchable encryption or in its asymmetric counterpart, trapdoor generation should be the exclusive ability of the search requester. Otherwise, any principal could launch a search, which will greatly aggravate the burden of the server. Thus, the trapdoor generation algorithm should use the private key of the search requester.

(2) The adversary could replace C_1 with any ciphertext of M' other than M without being noticed. For example, he/she could replace $C_1 = <y_{11}, y_{12}>$ with $C_1' = <y_{11}', y_{12}'>$, in which C_1' is the outcome of encrypting M' other than M with Bob's public key. Thus, the adversary could easily cause misunderstanding between Alice and Bob. For this, we should prevent C_1 from being tampered with, such as using hash functions or digital signature.

(3) All key pairs are generated by Alice. Usually, both the knowledge for security and the computing power of users are rather limited. Therefore, there might be various flaws in the generated key pairs, such as weak pseudorandomness, short length, and apparent semantics, etc. Therefore, in cryptographic schemes, key pairs are usually generated by Key Generator. Moreover, each principal computes under its own field, which incurs extra difficulty for the implementation of the scheme. Usually, for most cryptographic schemes, all computations could be done under just one field.

(4) Identities are encrypted. Generally speaking, all the identities are public. For an adversary capable of monitoring the whole communication, the sender and the receiver of the message could be known easily. Hence, it is unnecessary to encrypt identities.

(5) S is absolutely useless. In Check phase, after obtaining $H(W_1)$, the server does not need to check $S_1 = S_2^{H(W_1)}$. It could just compare $H(W_1)$ with $H(W)$ in sig. If $H(W_1) = H(W)$, it sends the corresponding D to Bob.

3 The Proposed Scheme

The proposed scheme contains 3 principals and 5 algorithms as well, shown as follows.

(1) Parameter generation

KG (Key Generator) selects a big prime p and a generator g of Z_p^* (All computations are done under this field, if unspecified.). KG selects a hash function $H : \{0,1\}^* \rightarrow Z_p^*$. KG generates key pairs $(sk_A, pk_A = g^{sk_A})$, $(sk_B, pk_B = g^{sk_B})$, $(sk_C, pk_C = g^{sk_C})$ for Alice, Bob and the server, respectively. KG sets the algorithm for encryption and signature to Elgamal.

(2) File sending

Alice computes $H(W)$ and $S = pk_B \cdot g^{H(W)}$.

Alice computes $C = Enc_{pk_B}(M) = \ < y_1, y_2 >$.

Alice computes $sig = Sig_{sk_A}(H(M)) = \ < H(M), r, s >$.

Alice sends S and $< C, sig, ID_A >$ to the server.

(3) Trapdoor generation

Bob sends $s = sk_B + H(W_1)$ to the server.

(4) File retrieving

The server computes $S' = g^s$, compares each S with S', and sends all $< C, sig, ID_A >$ where $S = S'$ to Bob.

(5) Verification

Bob fetches the public key pk_A corresponding to ID_A, which will be used to verify sig. If the verification succeeds, Bob decrypts C with sk_B to obtain M'. Bob computes $H(M')$ and discards once $H(M') \neq H(M)$.

4 Analyses of the Proposed Scheme

Apparently, the proposed scheme has perfectly fixed the aforementioned five flaws, shown as follows.

(1) Only the search requester could generate the trapdoor. From the trapdoor generation phase we could see, the private key of Bob, namely sk_B, is required, which is only possessed by Bob himself. Therefore, only the search requester is capable of generating the trapdoor.

(2) The adversary couldn't replace C with the ciphertext of M' other than M without being noticed. Suppose the adversary replaces C with $C' = Enc_{pk_B}(M') = \ < y_1', y_2' >$, let's discuss two cases below:

1) If the adversary replaces the 1st element in *sig* with $H(M')$, as he/she has no knowledge of the private key of Alice, namely sk_A, due to the unforgeability of Elgamal signature, he/she couldn't replace the 2nd and 3rd elements with Alice's legal signature r', s' on $H(M')$. Thus, in Verification phase, Bob definitely discards after verifying *sig* with Alice's public key pk_A.

2) If the adversary does not modify *sig* at all, then, in Verification phase, Bob will succeed when verifying *sig*. Then, after decrypting C' with sk_B, Bob will obtain M'. Due to collision resistance of $H(.)$, the probability of $H(M') = H(M)$ is negligible, which implies Bob will discard.

In a word, now the adversary could not cause any misunderstanding between Alice and Bob, which means the semantics of both principals is maintained.

(3) All key pairs are generated by KG, which ensures the security. All the computations are under the field Z_p^*, which avoids unnecessary troubles.

(4) Encryption of identities is removed. The identities of all principals are transmitted in the plaintext form, which avoids the overhead of encryption.

(5) There is a clear purpose for each component of the proposed scheme. None of them is useless, obviously.

5 Conclusion

This paper proposes an improved scheme aiming at fixing 5 flaws in the scheme proposed in [18]. Analyses show that the proposed scheme has well made up for the deficiency of the previous scheme and proves quite practical as well. In the future, we plan to design searchable encryption algorithms under more complicated application background, studying the issues including dynamic user group, single sender and multiple receivers, untrusted server, etc.

Acknowledgments. This research is financially supported by the Natural Science Foundation of China (No. 61462033). Thanks go to my supervisors Changxuan Wan & Zuowen Tan.

References

1. National Bureau of Standards. Data Encryption Standard, FIPS-Pub. 46. National Bureau of Standards, U. S. Department of Commerce, Washington D.C. (1977)
2. Daemon, J., Rijmen, V.: The Design of Rijndael: AES – The Advanced Encryption Standard. Springer, Heidelberg (2002)
3. Rivest, R., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM **21**(2), 120–126 (1978)
4. Song, XD., Wagner, D., Perrig, A.: Practical techniques for searches on encrypted data. In: Proceedings of IEEE Symposium on Security and Privacy, Oakland, USA, pp. 44–55. IEEE (2000)
5. Fang, L.: Research on Public Key Encryption with Keyword Search. Nanjing University of Aeronautics and Astronautics, Nanjing (2012)

6. Boneh, D., Crescenzo, G., Ostrovsky, R., Persiano, G.: Public key encryption with keyword search. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 506–522. Springer, Heidelberg (2004). doi:10.1007/978-3-540-24676-3_30
7. Abdalla, M., Bellare, M., Catalano, D., et al.: Searchable encryption revisited: consistency properties, relation to anonymous IBE, and extensions. J. Cryptology **21**(3), 350–391 (2008)
8. Khader, D.: Public key encryption with keyword search based on k-resilient IBE. In: Gavrilova, M., Gervasi, O., Kumar, V., Tan, C.J.K., Taniar, D., Laganá, A., Mun, Y., Choo, H. (eds.) ICCSA 2006. LNCS, vol. 3982, pp. 298–308. Springer, Heidelberg (2006). doi:10.1007/11751595_33
9. Crescenzo, G., Saraswat, V.: Public key encryption with searchable keywords based on jacobi symbols. In: Srinathan, K., Rangan, C.,Pandu, Yung, M. (eds.) INDOCRYPT 2007. LNCS, vol. 4859, pp. 282–296. Springer, Heidelberg (2007). doi:10.1007/978-3-540-77026-8_21
10. Hwang, Y.H., Lee, P.J.: Public key encryption with conjunctive keyword search and its extension to a multi-user system. In: Takagi, T., Okamoto, T., Okamoto, E., Okamoto, T. (eds.) Pairing 2007. LNCS, vol. 4575, pp. 2–22. Springer, Heidelberg (2007). doi:10.1007/978-3-540-73489-5_2
11. Baek, J., Safavi-Naini, R., Susilo, W.: Public key encryption with keyword search revisited. In: Gervasi, O., Murgante, B., Laganà, A., Taniar, D., Mun, Y., Gavrilova, Marina, L. (eds.) ICCSA 2008. LNCS, vol. 5072, pp. 1249–1259. Springer, Heidelberg (2008). doi:10.1007/978-3-540-69839-5_96
12. Rhee, H.S, Park, J.H, Susilo, W., et al.: Improved searchable public key encryption with designated tester. In: Proceedings of the 4th International Symposium on Information, Computer, and Communications Security, Sydney, Australia, pp. 376–379. ACM (2009)
13. Zhao, Y., Chen, X., Ma, H., et al.: A new trapdoor-indistinguishable public key encryption with keyword search. J. Wirel. Mobile Netw. Ubiquit. Comput. Dependable Appl. **3**(1/2), 72–81 (2012)
14. Yang, H., Xu, C., Zhao, H.: An efficient public key encryption with keyword scheme not using pairing. In: Proceedings of the 1st International Conference on Instrumentation, Measurement, Computer, Communication and Control, Beijing, China, pp. 900–904. IEEE (2011)
15. Luo, W., Tan, J.: Public key encryption with keyword search based on factoring. In: Proceedings of the 2nd International Conference on Cloud Computing and Intelligent Systems, Hangzhou, China, pp. 1245–1247. IEEE (2012)
16. Xia, Q., Ni, J., Kanpogninge, A., et al.: Searchable public-key encryption with data sharing in dynamic groups for mobile cloud storage. J. Univ. Comput. Sci. **21**(3), 440–453 (2015)
17. Shao, Z., Yang, B., Zhang, W., et al.: Secure medical information sharing in cloud computing. Technol. Health Care **23**, S133–S137 (2015)
18. Liu, P., Zu, L., Bai, C., et al.: A verifiable public key searchable encryption scheme. Comput. Eng. **40**(11), 118–120 (2014)